Designing for launchd

Best Practices and Design Patterns for Mac OS X

Session 500

These are confidential sessions—please refrain from streaming, blogging, or taking pictures.

At the convention center, a movie will be placed here

Designing for launchd

Best Practices and Design Patterns for Mac OS X

Damien Sorresso
Software Engineer

Introduction

Why launchd?

- Consolidated process management
- Data-driven
- Less code
- On-demand architecture

What You'll Learn

- Essentials of launch on demand
- Best design practices
- Session structure
- What's new in Snow Leopard

Quick Review

What is launchd?

- System bootstrapper (PID 1)
- Manages:
  - System jobs (daemons)
  - User jobs (agents)
- Canonical way of launching processes
- Virtual process manager

“All problems can be solved by introducing an extra level of indirection.”

Andrew Koenig, *The Fundamental Theorem of Software Engineering*

Jobs as Virtual Processes

Analogous to virtual memory

Load job	Map page
Start job	Fault page in
Stop job	Evict page
Unload job	Unmap page
Demand via IPC	Demand via read/write

Launch On Demand

Interprocess Communication (IPC)

- Jobs virtually available as IPC destinations
- Started in response to IPC request
- IPC mechanics hidden behind API

Why Launch On Demand?

Efficiency

Flexibility

Reliability

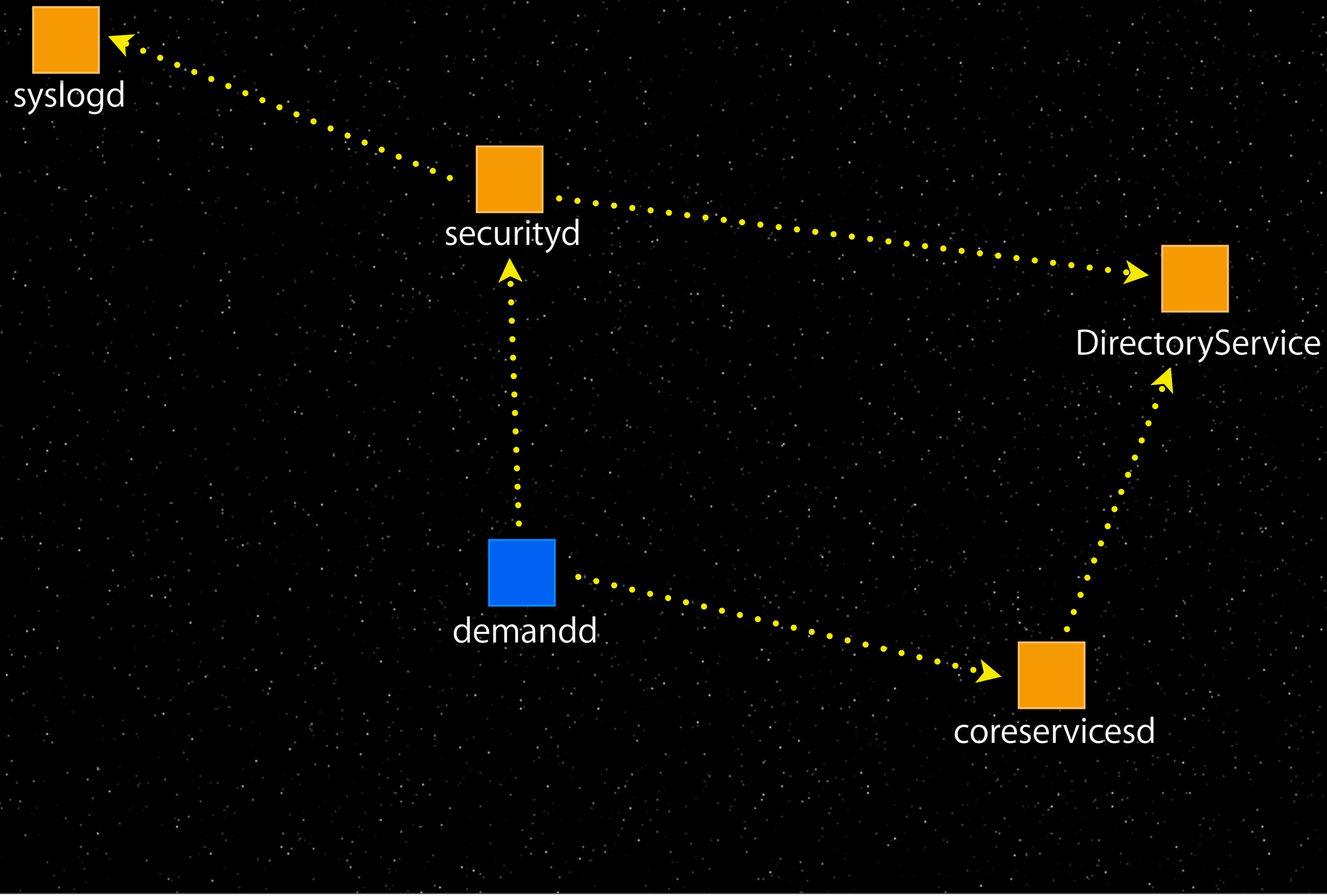
Why Does This Make Sense?

- History of on-demand virtualization
  - Main memory
  - GPU memory (CoreGraphics)
  - GPU resources (OpenCL, CoreImage)
  - Threads (NSOperation, Grand Central Dispatch)
- Standardized interface

Practical Applications

- Privilege separation
- Fault/resource isolation
- Dynamic response to state changes
- MVC for processes

launchd Job Basics



launchd Job Basics

Job dependencies

- More flexible
- Plugin architecture
  - Plugins run in separate address spaces
  - Just add a plist

The launchd Property List

Describing your job

- launchd property list describes…
  - Environment
  - Services
  - When/how to run
- ***Not*** for storing application preferences
- Treat as immutable

The launchd Property List

Loading and unloading jobs

- `launchctl load <PathToPlist>`
- `launchctl unload <PathToPlist>`

The launchd Property List

Required keys

- Label
  - Reverse-DNS
  - Must be unique
- Program
  - Path to executable
  - First argument to execve(2)
- ProgramArguments
  - Argument vector program receives
  - Second argument to execve(2)

The launchd Property List

KeepAlive

- Boolean or dictionary
- Criteria determine when job will run
  - File presence/absence
  - Successful/unsuccessful exit
  - Other job enabled/not enabled
- ORed together
- See launchd.plist(5) for more details

The launchd Property List

A simple daemon

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
     <key>Label</key>
     <string>com.apple.wwdc-sample</string>
     <key>Program</key>
     <string>/usr/libexec/wwdc-sampled</string>
     <key>ProgramArguments</key>
     <array>
          <string>wwdc-sampled</string>
          <string>-year</string>
          <string>2009</string>
     </array>
     <key>KeepAlive</key>
     <true/>
     <key>EnableTransactions</key>
     <true/>
</dict>
```

The launchd Property List

A simple daemon

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
     <key>Label</key>
     <string>com.apple.wwdc-sample</string>
     <key>Program</key>
     <string>/usr/libexec/wwdc-sampled</string>
     <key>ProgramArguments</key>
     <array>
          <string>wwdc-sampled</string>
          <string>-year</string>
          <string>2009</string>
     </array>
     <key>KeepAlive</key>
     <true/>
     <key>EnableTransactions</key>
     <true/>
</dict>
```

The launchd Property List

EnableTransactions

- Fast shutdown
- SIGTERM interrupts current transactions
- SIGKILL if no current transactions
- `vproc_transaction_begin()` and `vproc_transaction_end()`
- See vproc.h

Demo

Damien Sorresso
Software Engineer

Bootstrapping

“It’s not so hard to lift yourself by your bootstraps once you’re off the ground.”

Daniel B. Luten

Bootstrapping

Loading jobs

- Delegated to launchctl(1)
- Reads and loads property lists
- Launch criteria evaluated afterward

Bootstrapping

Disabled jobs

```
<key>Disabled</key>
<true/>
```

- Disabled jobs skipped when loading
- Property list contains default value
  - May be overridden elsewhere
  - Can be ignored with `launchctl load -f`

Bootstrapping

Disabled jobs

- `launchctl load -w <PathToPlist>`
- `launchctl unload -w <PathToPlist>`

Note: No longer modifies property list in Snow Leopard

Daemons

Systemwide background jobs

- Loaded during system startup
- Privileged execution
- No GUI interaction
- Singletons

/Library/LaunchDaemons/

/System/Library/LaunchDaemons/

Agents

Per user jobs

- Loaded during user login
- User execution
- GUI interaction
- One per user

~/Library/LaunchAgents/

/Library/LaunchAgents/

/System/Library/LaunchAgents/

Listing Jobs

- `launchctl list`
  - List all jobs
- `launchctl list <JobLabel>`
  - Information about specified job

- Use sudo for system launchd

Listing Jobs

Different job types

Job Type	Managed	Label Format
Normal	YES	com.mycompany.myproduct
Anonymous	NO	0xdeadbeef.anonymous.progname
LaunchServices	YES	[0x0-0xdeadbeef].com.mycompany.myproduct
mach_init	YES	0xdeadbeef.mach_init.progname

Listing Jobs

```
[durandal@Marathon]: ~$ launchctl list
PID      Status   Label
6978     -        0x10020a9c0.anonymous.launchctl
6938     -        0x10020a700.anonymous.bash
6937     -        0x10020a370.anonymous.login
6935     -        [0x0-0xd00d0].com.apple.Terminal
-        -        0x100200520.mach_init.rcd
6471     0        com.apple.Finder
218      -15      com.apple.Dock.agent
```

Listing Jobs

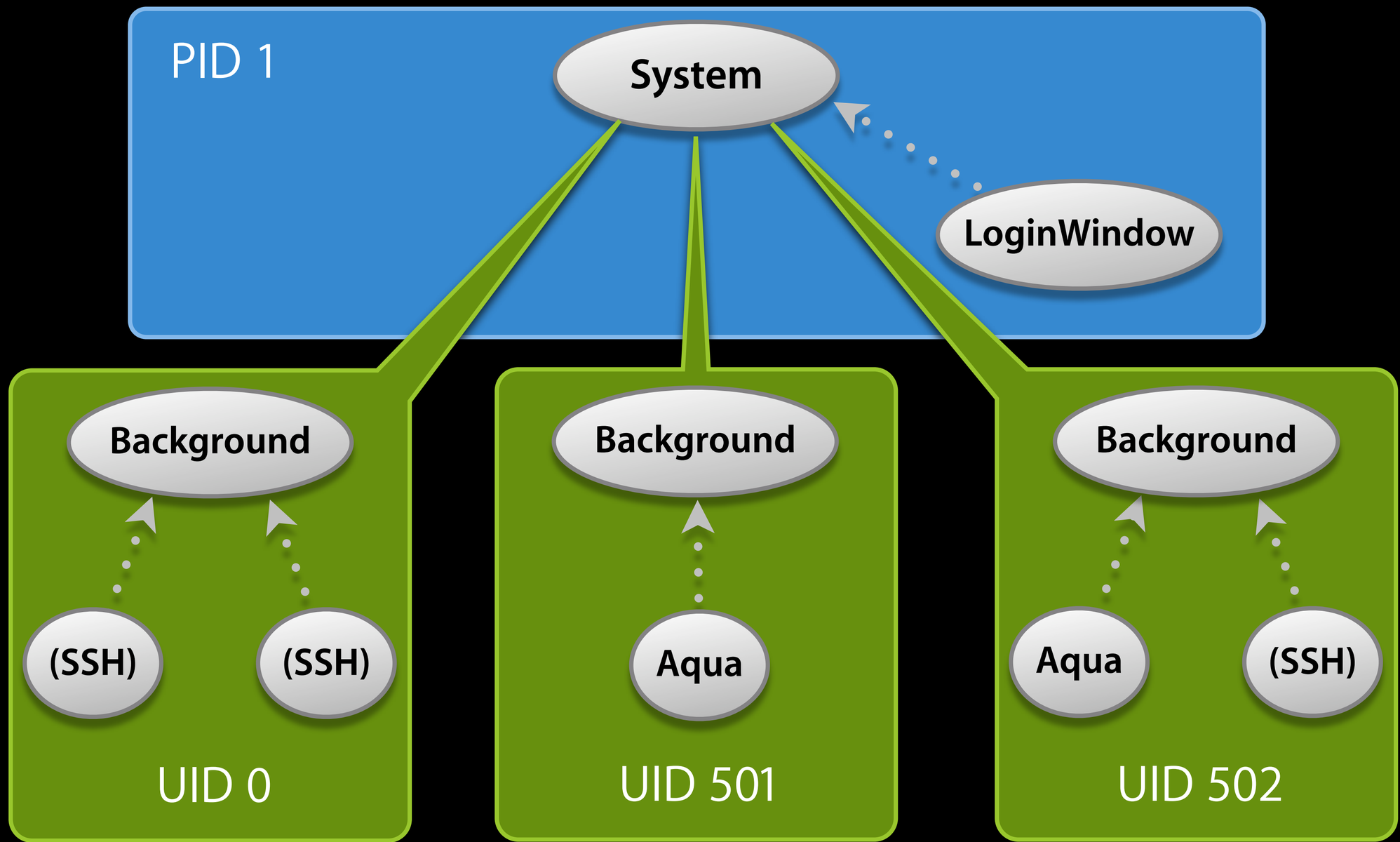
```
[durandal@Marathon]: ~$ launchctl list com.apple.Finder
{
	"Label" = "com.apple.Finder";
	"LimitLoadToSessionType" = "Aqua";
	"OnDemand" = true;
	"LastExitStatus" = 0;
	"PID" = 6471;
	"TimeOut" = 30;
	"Program" = "/System/Library/CoreServices/Finder.app/Contents/MacOS/Finder";
	"EnableTransactions" = true;
	"TransactionCount" = 0;
	"MachServices" = {
		"com.apple.finder.ServiceProvider" = mach-port-object;
	};
	"PerJobMachServices" = {
		"com.apple.FileSyncAgent.callbacks" = mach-port-object;
		"com.apple.CFPasteboardClient" = mach-port-object;
		"com.apple.coredrag" = mach-port-object;
		"com.apple.tsm.portname" = mach-port-object;
		"com.apple.axserver" = mach-port-object;
		"WakeUpProcessPort" = mach-port-object;
	};
};
```

ServiceManagement Framework

- CoreFoundation-layer API
- `SMJobCopyDictionary()` and more…
- Allows interaction with system launchd via AuthorizationRef

Sessions

Sample Session Layout



launchd Sessions

Your job's environment

- Combination of job set and Mach bootstrap
- Arranged in a hierarchical fashion
  - Mach service lookups go up
- `sudo launchctl bstree` New
- `launchctl bslist`

launchd Sessions

Your job's environment

- System—for daemons
  - Hosted by PID 1
  - Root Mach bootstrap
- Per user—for agents
  - Hosted in per user launchd's
  - User-private Mach bootstrap

launchd Sessions

Targeting sessions

- Agents can specify session
- Default to Aqua session
- Use `LimitLoadToSessionType` to override

```
<key>LimitLoadToSessionType</key>
<string>Background</string>
```

launchd Sessions

User vs. UID

- User is more than just POSIX bits (UID, GID, etc.)
- launchd provides canonical user environment
- Unifies POSIX and Mach aspects of user

launchd Sessions

UID 0's per user launchd

- Login as root runs as root user
- Daemons run with UID 0 but run as system
- Root shell interacts with system (PID 1) launchd
- Consistent with previous releases
- `launchctl -u root <subcommand>`

launchd Sessions

Debugging environment

- Don’t launch with gdb

```
<key>WaitForDebugger</key>
<true/>
```

- Stalls job at first instruction so you can attach
- `gdb attach --waitfor mydaemond`
- Transaction APIs require launchd
- Debug and run under same environment!

launchd Sessions

Mach bootstrap hierarchy in per-user sessions

- Flat for user sessions
- GUI and SSH get same services
- SSH login equivalent to Terminal.app login
- Not X11-style remote display
- Makes screen(1) much happier

Demo

Damien Sorresso
Software Engineer

About Those Session Things…

- CoreGraphics, Security, launchd sessions overlap
- Tied together via task bootstrap port
- Implementation detail
- Should not affect majority of developers

Architectural Design

Architecturally Speaking

Privilege separation

- Daemons and agents work together
- Common Unix design pattern
  - Daemon listens on privileged port
  - Spawns child for new request
  - Child drops privileges, handles request
- Convert child to agent

Architecturally Speaking

Privilege separation

- New architecture
  - Daemon listens on privileged port
  - Agent checks in with daemon, vends IPC resource for communication
  - Daemon launches agent on demand, gets data
- Agent spawned in user's environment

Architecturally Speaking

Privilege escalation

- GUI applications run unprivileged
- Need separate daemon to perform privileged tasks
- Installing daemon itself requires privileges

Architecturally Speaking

Privilege escalation

- New API: `SMJobBless()`

  - Automates installation of privileged helper daemons
  - Leverages code signing to install securely
  - launchd launches daemon on demand

Managing User Privileges and Operations with Authorization Services	North Beach Wednesday 3:30 PM
Assigning Your Application an Identity with Code Signing	Nob Hill Tuesday 9:00 AM

Architecturally Speaking

Fault and resource isolation

- Safari on Snow Leopard
  - WebKit plug-ins launched on demand
  - Drawing into Safari done entirely through IPC
- Fault and resource isolation
  - Plug-in crashes don’t affect Safari
  - Use no resources if not needed

Demo

Damien Sorresso
Software Engineer

Wrapping Up

What have we covered today?

- Basic launchd philosophy—virtual processes, launch on demand
- How daemons and agents interact
- Leveraging daemons and agents in your architecture
- What's been added in Snow Leopard

For More Information

Website and Sources
http://launchd.macosforge.org

Email Discussion List
launchd-dev@lists.macosforge.org

Documentation
launchd(8)
launchd.plist(5)
launchctl(1)
ServiceManagement/ServiceManagement.h

For More Information

Managing User Privileges and Operations with Authorization Services	North Beach Wednesday 3:30 PM
Assigning Your Application an Identity with Code Signing	Nob Hill Tuesday 9:00 AM

Q&A

WWDC Session Survey

You can rate your sessions on the WWDC Attendee Site.

Survey forms are located on each session description page at:

developer.apple.com/wwdc/attendee

WWDC09